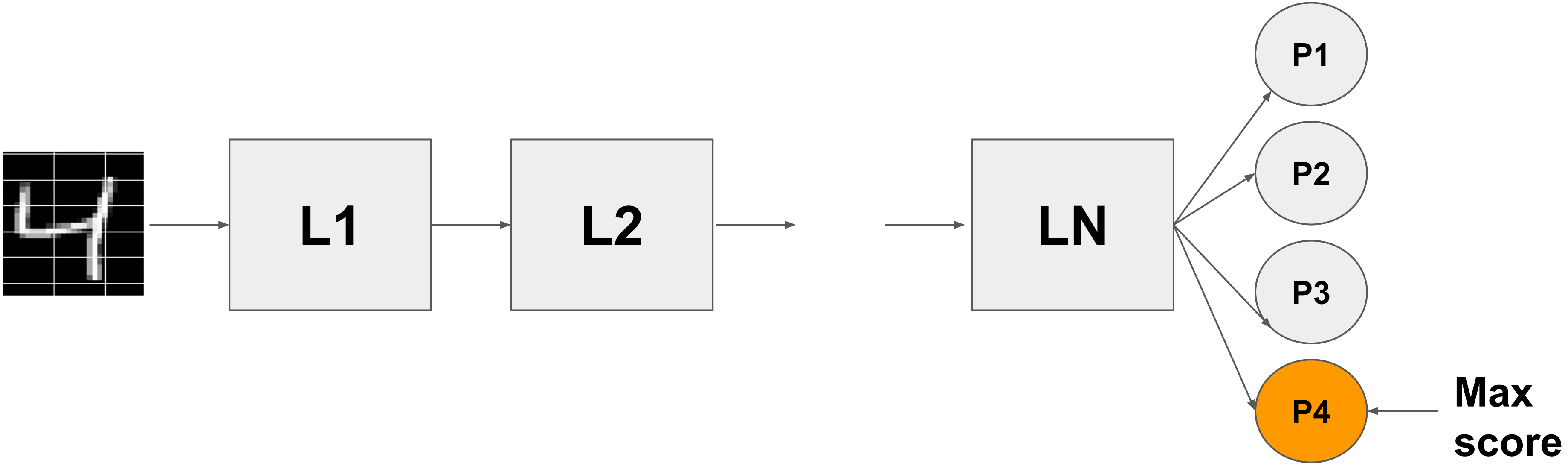
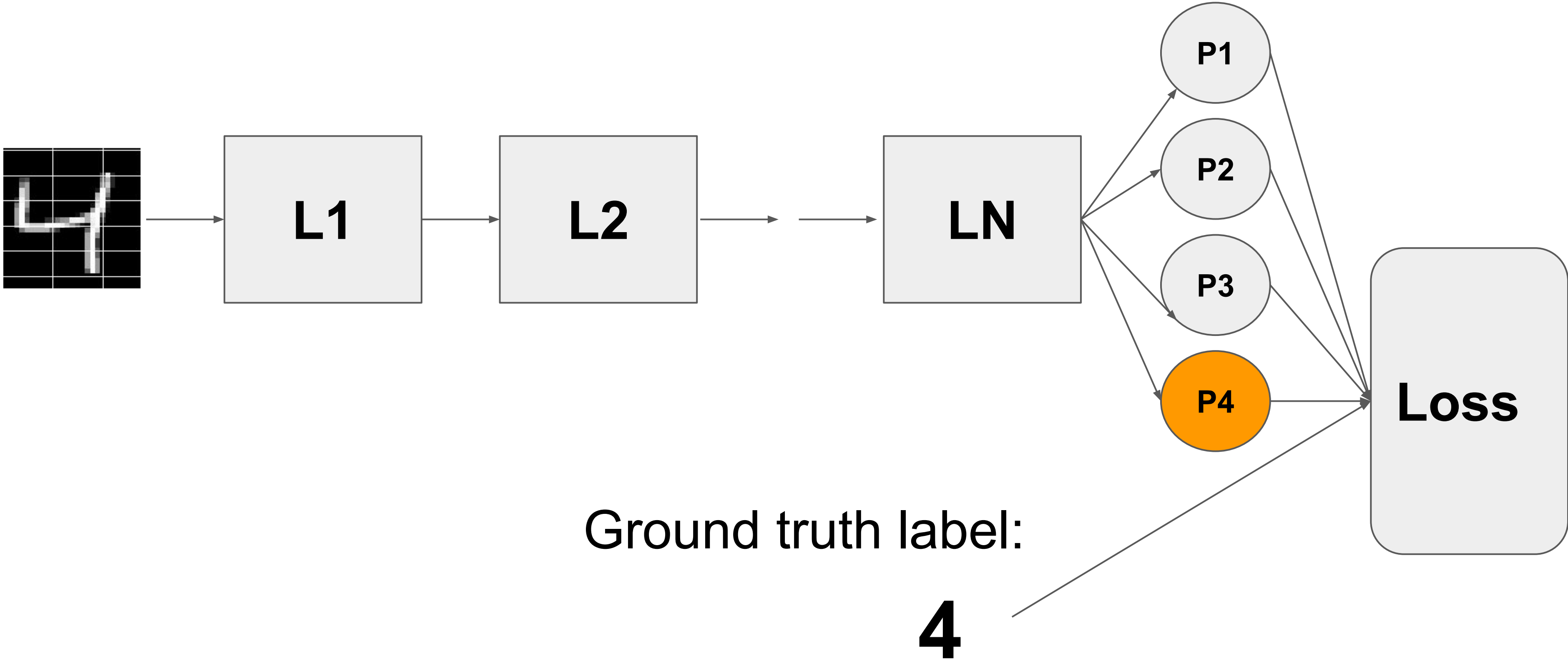
Backpropagation reminder+

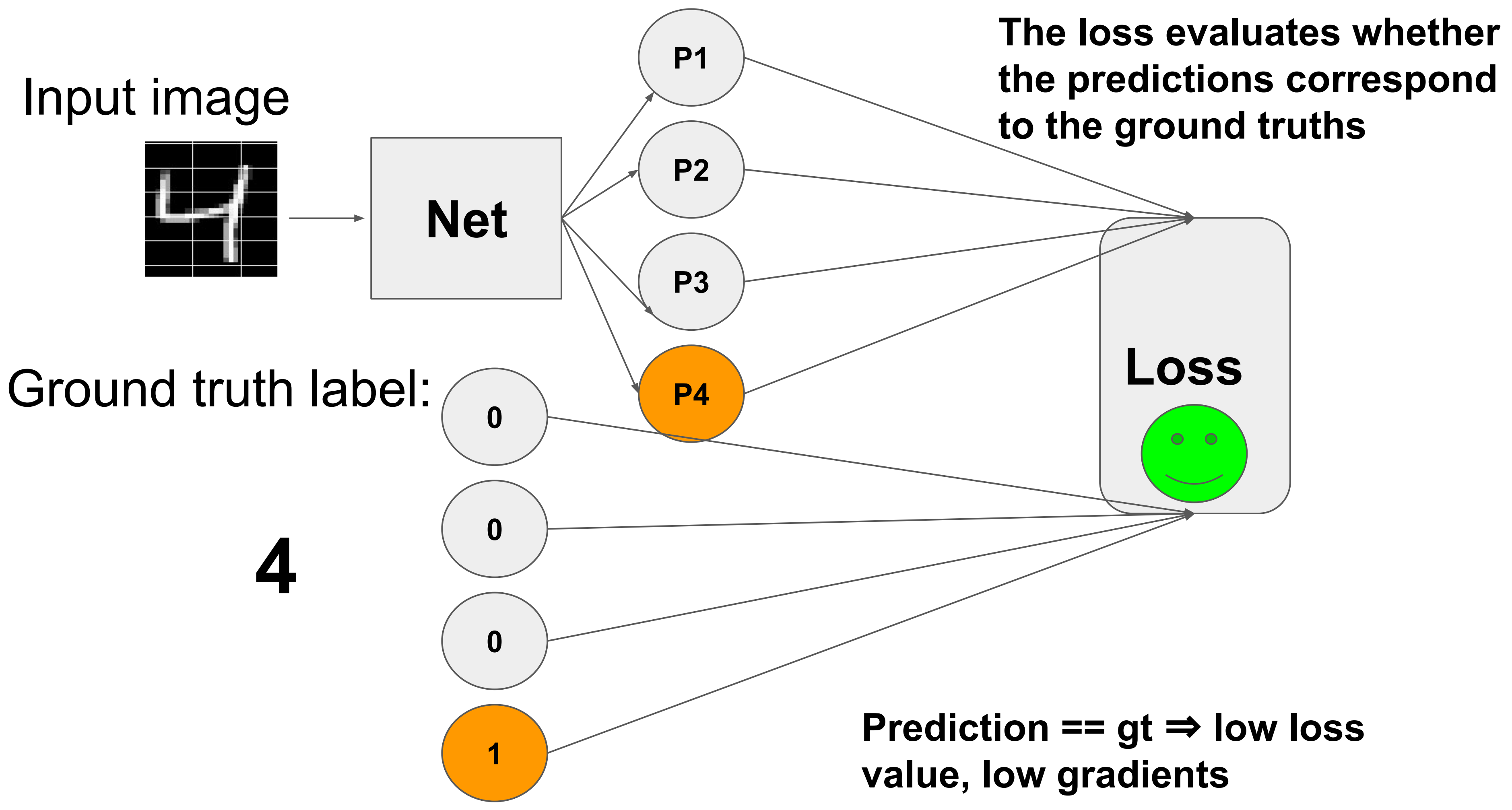
29.03.19

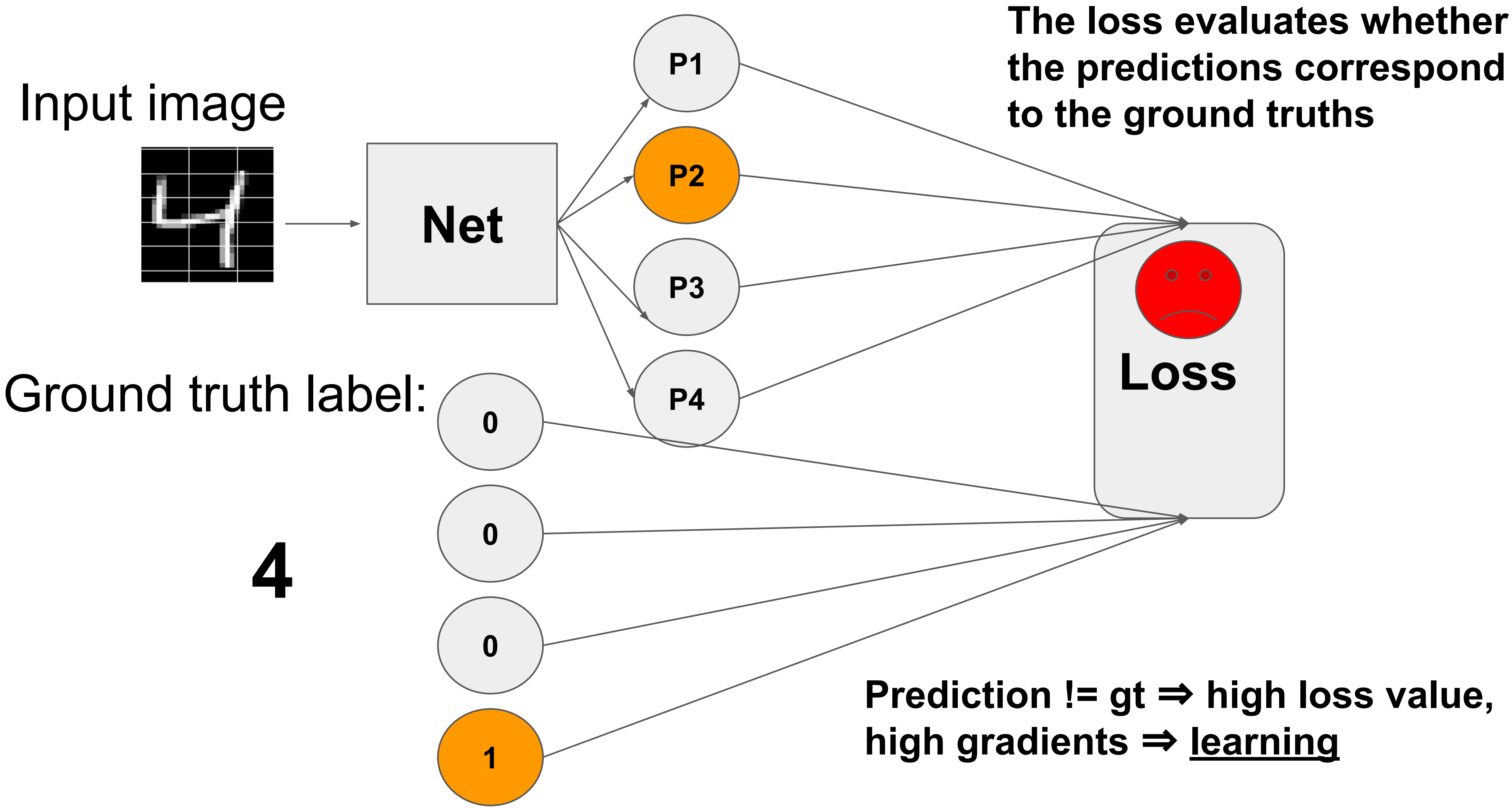
Neural network for multiclass classification



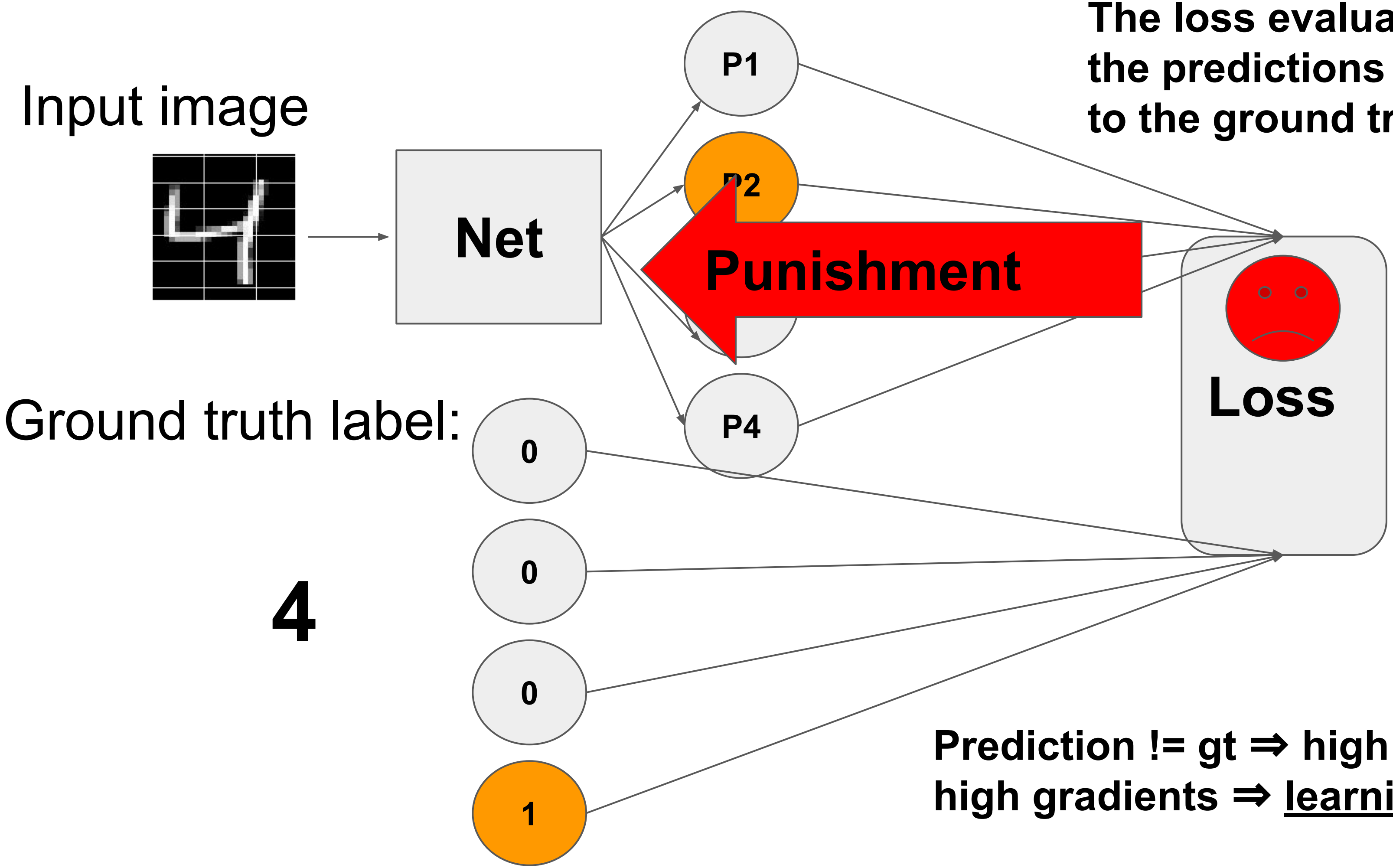
Neural network for multiclass classification: training




Input image
Net
P1
P2
P3
P4
The loss evaluates whether the predictions correspond to the ground truths
Loss
Ground truth label:
4
0
0
0
1
Prediction == gt ⇒ low loss value, low gradients


The loss evaluates whether the predictions correspond to the ground truths
Input image
Net
P1
P2
P3
P4
Loss
Ground truth label:
4
0
0
0
1
Prediction != gt ⇒ high loss value, high gradients ⇒ learning

The loss evaluates whether the predictions correspond to the ground truths



Prediction != gt ⇒ high loss value, high gradients ⇒ <u>learning</u>

Example: log Softmax (used in the code)

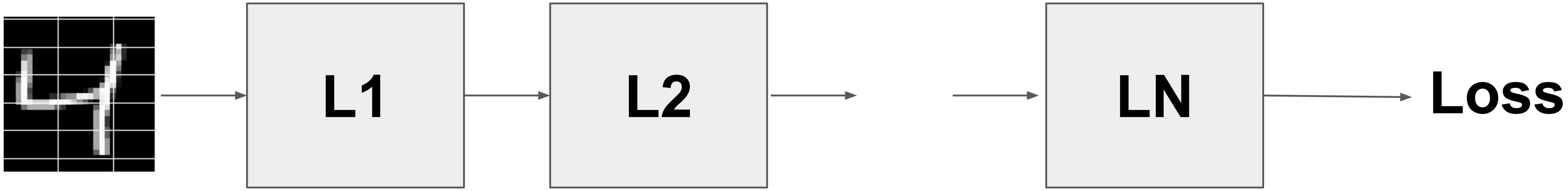
$$loss = -log \frac{e^{a_{correct}}}{\sum_i e^{a_i}}$$

$$loss = -a_{correct} + log \sum_i e^{a_i}$$

$$\frac{\partial loss}{\partial a_i} = \begin{cases} \frac{e^{a_i}}{\sum_k e^{a_k}} & \text{if } i \neq \text{correct} \\ \frac{e^{a_i}}{\sum_k e^{a_k}} - 1 & \text{if } i = \text{correct} \end{cases}$$

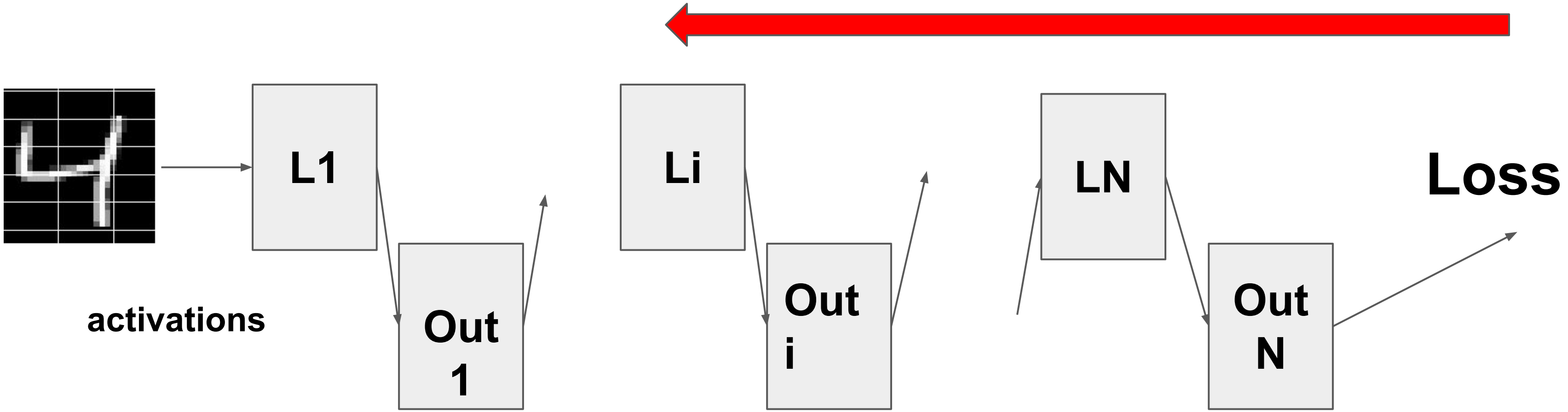
If our prediction is close to ideal => gradients are close to zero

Backpropagation: calculate $\dfrac{\partial loss}{\partial w}$ for every parameter w



$\dfrac{\partial loss}{\partial W_1}$? $\dfrac{\partial loss}{\partial W_2}$? $\dfrac{\partial loss}{\partial W_N}$?

How to calculate $\frac{\partial loss}{\partial w}$



$$loss = loss(Out_N) = loss(Out_N(L_n(Out_{N-1}(L_{N-1}(...Out_i(L_i(Out_{i-1}(...); W_i))...); W_{N-1})); W_N))$$

$$\boxed{\frac{\partial loss}{\partial W_i} = \frac{\partial loss}{\partial Out_N} \frac{\partial Out_N}{\partial Out_{N-1}} \cdots \frac{\partial Out_{i+1}}{\partial Out_i} \frac{\partial Out_i}{\partial W_i}}$$

Backprop

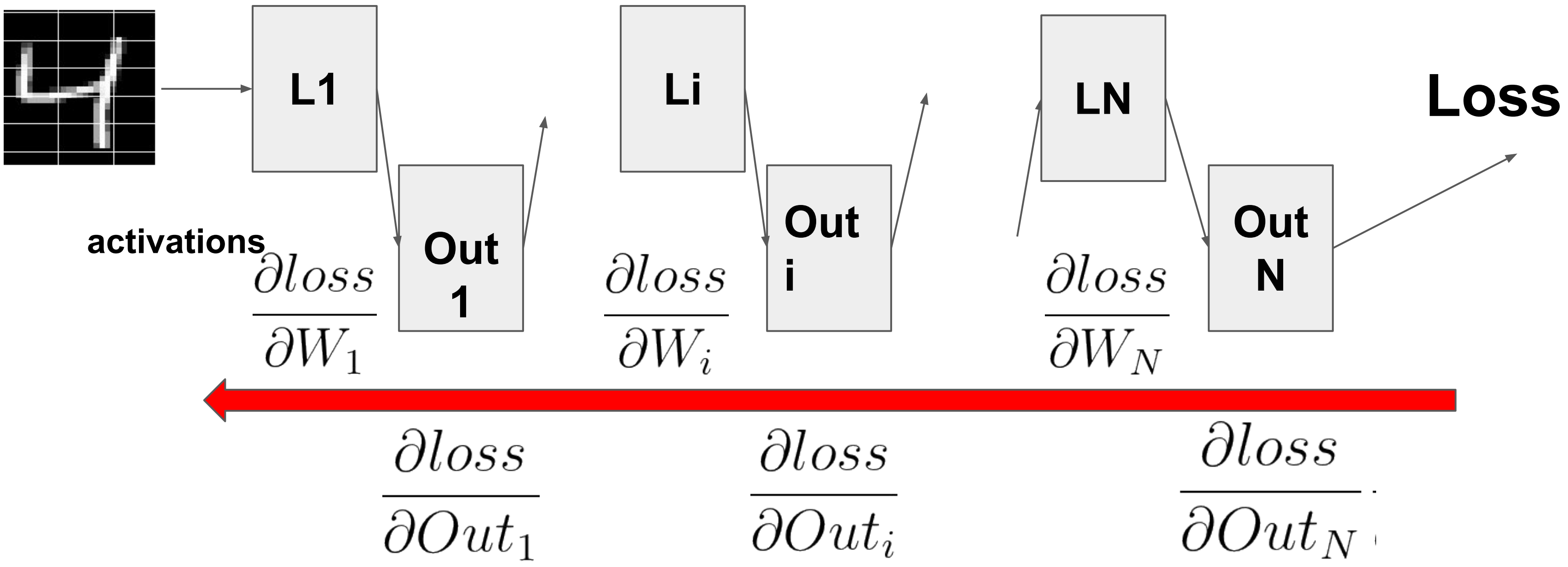
From the last layer to first :

$$\frac{\partial loss}{\partial Out_i} = \frac{\partial loss}{\partial Out_{i+1}} \frac{\partial Out_{i+1}}{\partial Out_i}$$

Known from the
previous iteration

$$\frac{\partial loss}{\partial W_i} = \frac{\partial loss}{\partial Out_i} \frac{\partial Out_i}{\partial W_i}$$

Backprop


L1
Li
LN
Loss
activations
Out 1
Out i
Out N
$\frac{\partial loss}{\partial W_1}$
$\frac{\partial loss}{\partial W_i}$
$\frac{\partial loss}{\partial W_N}$
$\frac{\partial loss}{\partial Out_1}$
$\frac{\partial loss}{\partial Out_i}$
$\frac{\partial loss}{\partial Out_N}$

Example: dense layer

$$Y = XW + b$$

We have

$$Y - N \times out$$

$$X - N \times in$$

$$W - in \times out$$

$$b - out$$

$$\frac{\partial loss}{\partial Y} - N \times out$$

We need to compute:

$$\frac{\partial loss}{\partial X} - N \times in$$

$$\frac{\partial loss}{\partial W} - in \times out$$

$$\frac{\partial loss}{\partial b} - out$$

Try to guess using sizes,
Consider 1dim case: y = xw + b

Dense layer: answers

$$Y = XW + b$$

We have

$$Y - N \times out$$

$$X - N \times in$$

$$W - in \times out$$

$$b - out$$

$$\frac{\partial loss}{\partial Y} - N \times out$$

We need to compute:

$$\frac{\partial loss}{\partial X} = \frac{\partial loss}{\partial Y} W^T$$

$$\frac{\partial loss}{\partial W} = X^T \frac{\partial loss}{\partial Y}$$

$$\frac{\partial loss}{\partial b} = \sum_{i}^{N} \frac{\partial loss}{\partial Y_{ik}}$$

Dense layer: derivation

$$\frac{\partial loss}{\partial X} = \frac{\partial loss}{\partial Y} W^T$$

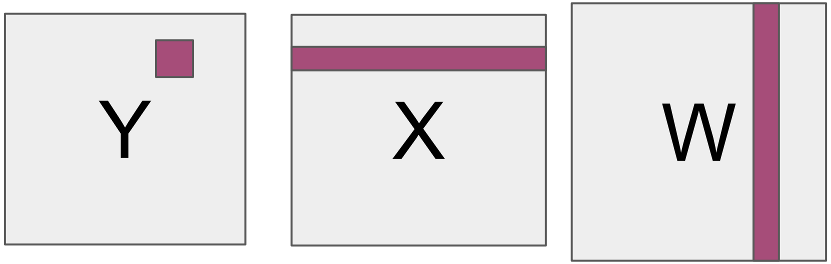
Y_i only depends on X_i !!!
(examples in the batch are independent)

$$[\frac{\partial loss}{\partial X}]_{ij} = \frac{\partial loss}{\partial X_{ij}} =$$

$$\sum_{k,l}^{N,out} \frac{\partial loss}{\partial Y_{kl}} \frac{\partial Y_{kl}}{\partial X_{ij}} =$$

$$\sum_{l}^{out} \frac{\partial loss}{\partial Y_{il}} \frac{\partial Y_{il}}{\partial X_{ij}} =$$

$$\sum_{l}^{out} \frac{\partial loss}{\partial Y_{il}} W_{jl}$$



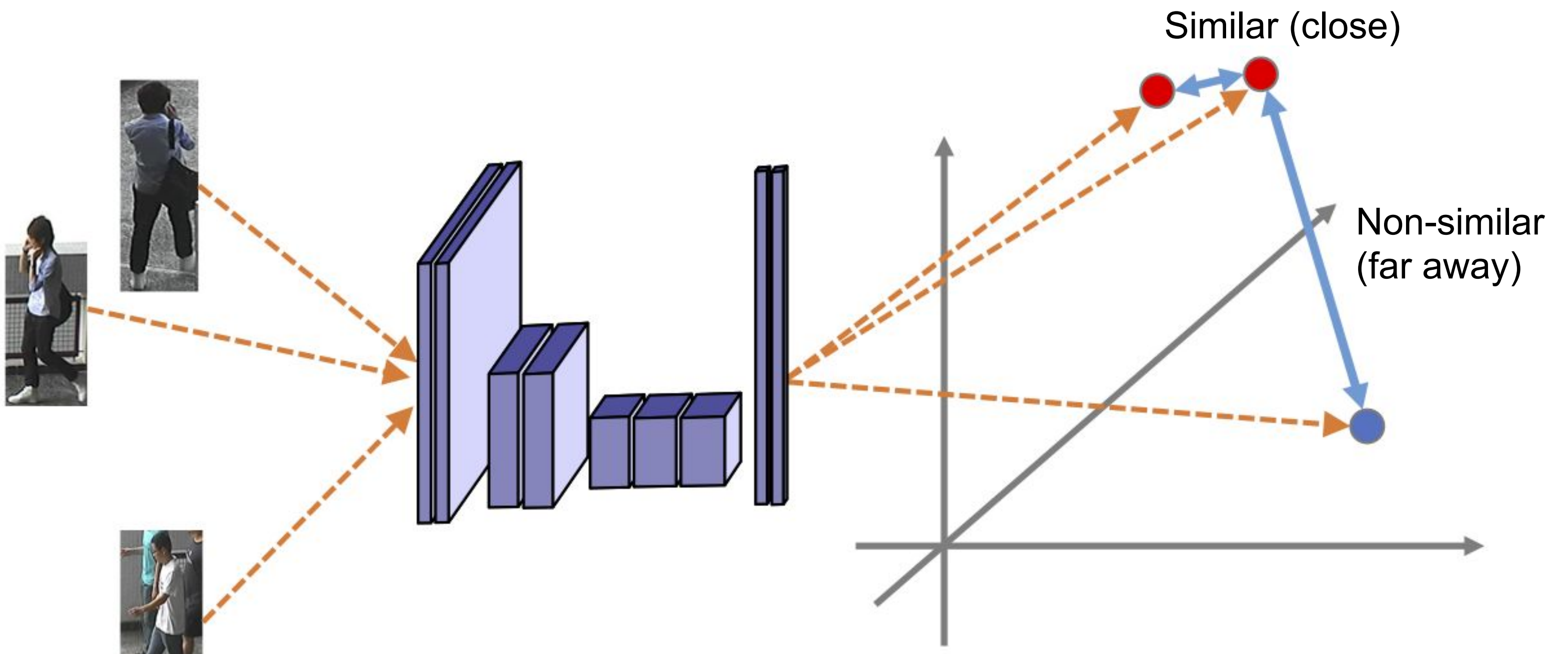
Beyond classification: image retrieval

- For each query image to find similar images (of the same class) in a gallery set
- Image retrieval vs classification: images of unseen classes are possible
- **The key problem** is to be able to **estimate similarity** value of image pairs so that all the gallery images could be sorted by their similarity to a query image

Query image

Matching gallery images

Deep learning for image retrieval



How to learn this?

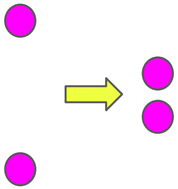
Input images　Deep Neural Network　Descriptor space

Contrastive loss

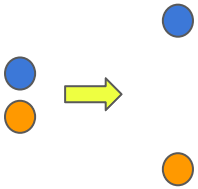
$$L_c(x_i, x_j, l) = \begin{cases} \frac{1}{N_{pos}} \|x_i - x_j\|_2^2 & l = 1 \\ \frac{1}{N_{neg}} (\max(0, M - \|x_i - x_j\|_2))^2 & l = 0 \end{cases}$$

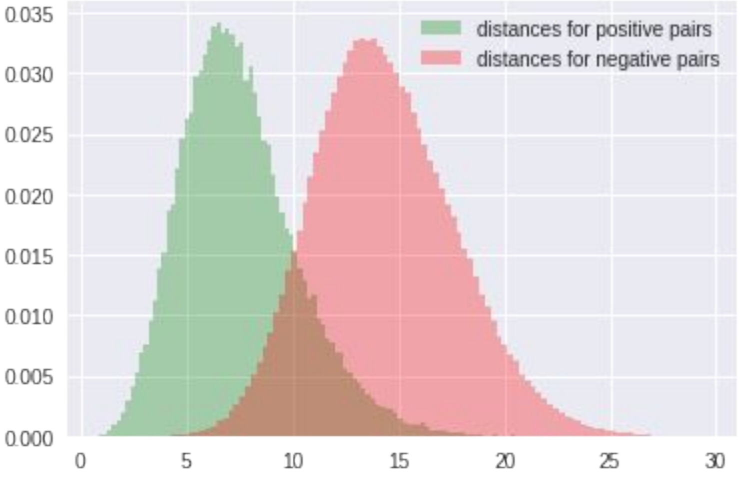
Changes in the descriptor space:

Similar: make closer

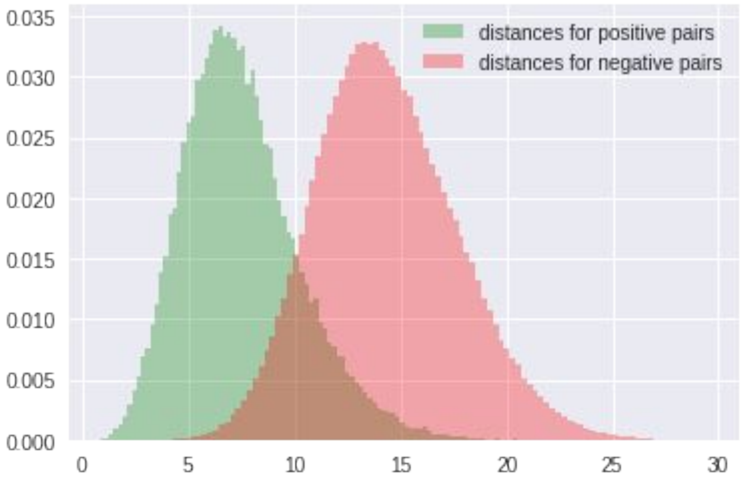


Non-similar: make farther





Distributions of the distance values



Often, less overlap mean better retrieval
(but of course, there are more direct retrieval metrics)

$$BC(p, q) = \sum(\sqrt{p_i q_i})$$

Bhattacharyya coefficient measures the overlap

(see example in the seminar code)